WEdge Scrum Process

13 JANUARY 2010

DOCUMENT VERSION CONTROL: 1.0

Revision History

Date	Version	Description
5 April 2010	1.0	Updated section A1.3 for risk assessment at sprint planning
22 February 2010	1.0	Updated section 3.6.3; process for bugs worked in sprint
13 January 2010	1.0	Editorial cleanup
11 January 2010	Final Draft	Incorporated team member input
4 January 2010	Draft	Initial copy

Table of Contents

List of Illustrations and Tables

This page intentionally left blank

About WEdge Software Development and Scrum

The Warfighter's Edge (WEdge) team is focused to become an industry leader in agile development methodology and high quality software. *Scrum* is an iterative, incremental framework for managing complex product development and is often a part of agile development.

WEdge uses Scrum methodology and primarily follows the Enterprise Content Management (ECM) templates in Microsoft Team Foundation Server (TFS). For additional details about Scrum, see http://www.mountaingoatsoftware.com/Scrum.

About This Document

This document describes the Scrum process followed by WEdge and is integral to compliance with that process. The document acts as the front line reference for team members who need to learn or refresh their knowledge of the Scrum tasks and conventions that are followed during planning, development, test, and release of WEdge software products.

Scrum methodology allows a process to be modified at the end of a development period, if necessary. Therefore, this is a living document that is periodically updated to reflect any process changes. Whenever updated, the document is delivered to the entire WEdge team, with change reasons summarized in the revision history in reverse chronological order.

The latest version of this document is available electronically on the team SharePoint site (currently http://portal) under *Team Documents* on the home page.

Intended Audience

All WEdge team members must become acquainted with this document and follow the procedures it describes. Some familiarity with general software development is assumed.

This page intentionally left blank

1 Scrum Process Overview

WEdge can take an idea from any source and formalize it into a Change Request (CR) as shown in Figure 1–1. A CR is refined into codifiable requirements, and test procedures are created to fulfill the requirements. All requirements from every CR are prioritized into a single list for development. Developers draw from the top of this list and determine the specific tasks needed to satisfy the requirement. Then, they create the code associated with the tasks.

When coding is complete, test scenarios are run by Software Quality Assurance (SQA) and, if passed, the requirement is completed. Once requirements are complete, the final feature is demonstrated internally and externally, where customers can give feedback and developers can make modifications during the development cycle. When ready, features are approved by the WEdge senior staff and scheduled for release to the user community.

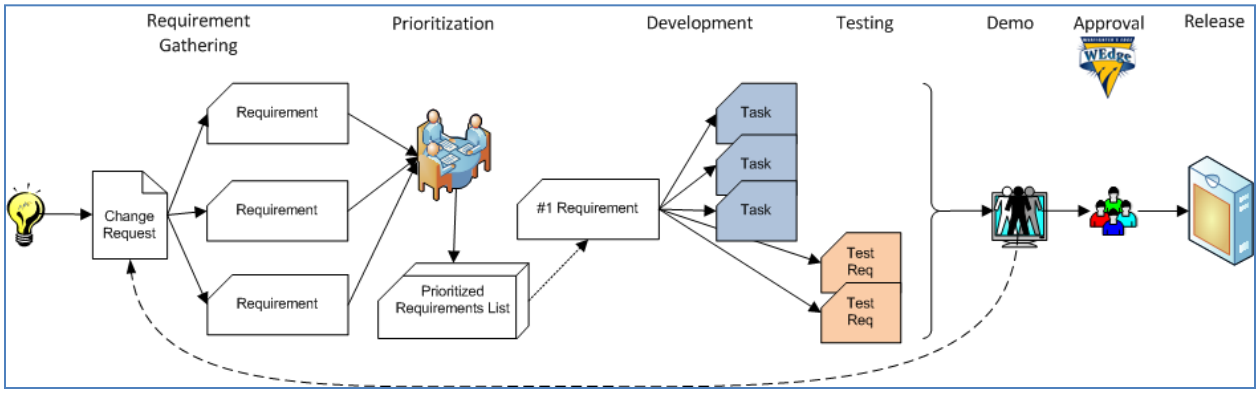

Figure 1–1 Scrum Process Overview

1.1 Scrum Terminology

The following terms relate to the timing and flow of events during the WEdge Scrum process.

1.1.1 Phase

WEdge releases software in a quarterly timeframe called a *phase*, with each phase broken down into smaller timeframes called sprints. A phase consists of four sprints, followed by a one-week phase planning period. The total phase length equates to a quarter of a year.

For more about phases, see Section 1.3.

1.1.2 Sprint

All software development occurs during a three week period called a *sprint*. Sprints start on a Wednesday and end on a Tuesday regardless of holidays.

For more about sprints, see Section 1.4.

1.1.3 Change Request

A Change Request (CR) marks the formal process to begin a new product, or modify or add functionality or ideas in an existing product. Functionality may include visible changes in behavior as well as transparent changes such as code refactoring or performance enhancements. Anyone can input a CR for any reason.

For more about CRs, see Section 3.1.

1.1.4 Product Backlog Item

Each CR is divided into manageable pieces called Product Backlog Items (PBIs). A PBI is scoped to be something that a single developer can complete within one sprint. PBIs are the focal point of the Scrum process, as they are the "requirements" from the CRs, and they track to corresponding Acceptance Tests (ATs). As such, PBIs are sometimes referred to as WEdge requirements.

For more about PBIs, see Section 3.2.

1.1.5 Sprint Backlog Task

Sprint Backlog Tasks (SBTs) are the tasks derived from each PBI, as analyzed during the sprint planning meeting by individual developers, and during subsequent design meetings for the PBI. SBTs are scoped to be something that a single developer can complete in under 5 hours.

For more about SBTs, see Section 3.3.

1.1.6 Acceptance Test

An Acceptance Test (AT) is a specific test written against a PBI and its Conditions of Acceptance (CoA) as evaluated by the SQA team.

For more about ATs, see Section 3.5.

1.1.7 Velocity

Velocity is a running average of estimation points per Scrum team, calculated by adding all estimated PBIs completed by each team and then dividing by the number of sprints completed thus far.

1.2 Team Roles and Personnel

The WEdge Scrum process involves the following personnel.

1.2.1 Scrum Team

A WEdge Scrum team consists of a Project Engineer, three to four developers, and a tester. WEdge has two full Scrum teams: *Columbia* and *Atlantis*.

Complementing the Scrum teams are technical documentation writers, graphic artists, a software architect, a deployment developer, and Certification and Accreditation (C&A) personnel.

1.2.1.1 Project Engineer

The Project Engineer (PE) is a certified ScrumMaster who oversees and ensures compliance with the WEdge Scrum process. The PE takes ownership of all PBIs, removes impediments, and ensures proper estimation of each PBI. Each PE is an expert in one or more WEdge products and works directly with the Product Owner (PO) of those projects.

1.2.1.2 Developer

The developer takes ownership of and implements SBTs. The developer also performs unit testing as part of test-driven development.

1.2.1.3 Tester

After unit tests are run, the tester validates the CoA of a PBI. Testers not only fully test the required functionality but also perform scenario based testing to fully complement testing.

1.2.2 Additional Roles and Personnel

The following personnel support the WEdge Scrum process, although they are not part of the formal Scrum team.

1.2.2.1 Product Owner

Each product WEdge maintains has a Product Owner (PO). This is normally a person in the operations division who is responsible for feature prioritization and overall acceptance of developed code.

1.2.2.2 Certification and Accreditation

The network and security division includes an Information Assurance Manager (IAM) who is responsible for Department of Defense (DoD) network compliance, Configuration Control Board (CCB) and the C&A process, as well as all DoD regulations that apply. Additionally, an Information Assurance Officer (IAO) is responsible for Security Technical Implementation Guide (STIG) review and compliance, Information Assurance Vulnerability Alert (IAVA) compliance, and application security procedures including Application Security Assurance Center of Excellence (ASACoE) vulnerability scans.

1.2.2.3 Architect

The architect has knowledge of the high level coding structure of every WEdge product and is responsible for integration of PBIs by providing appropriate interfaces and structure. This person is also responsible for interaction between WEdge products.

1.2.2.4 Deployment Developer

The deployment developer is responsible for all software installers and upgrade procedures for each WEdge product.

1.2.2.5 Product Development Leads

The product development lead is a developer with oversight of a particular product feature within the WEdge applications. This person is involved with code reviews and the design of the product whenever it affects his or her area of expertise.

1.2.2.6 Funding Stakeholders

WEdge has interested parties (e.g., Air Combat Command, Air National Guard, Air Staff) who sponsor development by contributing annual funding. The stakeholders are given a quarterly update on progress from the WEdge Director and are invited to a yearly Project Management Review (PMR), normally held in July.

1.2.2.7 Users

WEdge users are warriors who voluntarily put their lives on the line for our Country and freedom. A unique benefit to WEdge is its inclusion of customers during the development process, when they are invited to feature demonstrations at the end of each sprint. The POs are the primary interface with customers and act as their designated proxies; however, anyone on the WEdge team can work with customers at any time. WEdge team members treat customers with the utmost respect whenever they are working together to improve the WEdge product.

1.2.2.8 Red Team

WEdge employs developers who are Certified Ethical Hackers (CEHs). As such, they have the duty to identify and exploit vulnerabilities in WEdge software in order to ensure integrity and security of WEdge products.

1.3 Phases

A phase is thoroughly planned before implementation. Once under way, metrics are presented to management throughout the phase in order to determine continuous release feasibility. Afterward, the phase is then reviewed for lessons learned and process modifications during the phase planning week.

A phase consists of four sprints and a phase planning week as shown in Figure 1–2 and Figure 1–3.

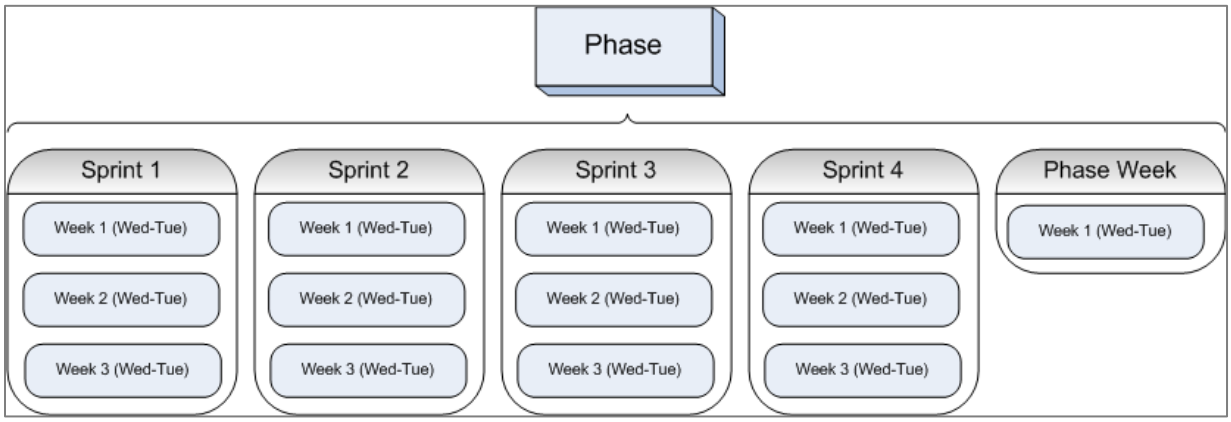

Figure 1–2 Phase Structure

MONDAY	TUESDAY	WEDNESDAY	THURSDAY	FRIDAY
	(Final Sprint Ends)	Day 1 • 0900 - Phase Retrospective • 1300 - Phase Planning	Day 2 • 0900 - Directors Meeting, phase plan concurrence	Day 3
Day 4	Day 5 • 1000 - CCB	(Next Phase Begins)		

Figure 1–3 *Phase Week* Timeline

1.3.1 Planning

To be successful, a phase must be planned. This is a fairly simple process when PBIs are fully estimated and the PO has prioritized CRs. For planning purposes, POs are expected to maintain an individual priority list for their own product, which also helps the Operations Director assemble the presentation for the weekly directors meeting.

1.3.1.1 Phase Planning Week

For the phase planning meeting, a combined list of all products is prioritized.

CRs are planned and prioritized as determined by the PO. Note that unforeseen circumstances and impediments may force certain features to be cut or added before the end of the phase.

The phase planning meeting is run by the Director of Application Engineering, and looks at each feature desired for inclusion in the next phase. Each CR desired for the phase must have its PBIs estimated before planning can begin. It is the responsibility of the PE for the product to ensure that CRs are broken up into PBIs and that each PBI has been appropriately estimated.

The WEdge team starts with a certain capacity to implement code during the next phase. As each CR is added to the phase, the capacity decreases. This gradual reduction is monitored to ensure that the development team is not overtasked for each sprint and phase.

Additional features or changes may be added or deleted from the prioritized CR list at the discretion of the PO during the phase, but never when it affects the current sprint.

During the phase planning week, when they are not creating code associated with a sprint, the development manager may task developers to perform prototyping, investigate bugs, review designs, or do whatever else is needed. However, developers are not allowed to work on sprint items or PBIs.

1.3.1.2 Planning Estimation

For PBI estimation, a PE can call an ad hoc session to ensure that PBIs are estimated for planning purposes. In this meeting, developers analyze the relative "bigness" of a PBI by gut feel and come to a consensus.

> **Note**
> The idea of PBI estimation is *not* to figure out exactly how much time a PBI will take, but to determine how big a PBI is relative to other PBIs.

This estimation is exact enough to adequately plan sprints and phases without fully determining how work will actually be done, which is accomplished later, in the sprint commitment meeting.

1.3.2 Implementation

During phase execution, management receives metrics updates in order to oversee progress toward product release. It is a management decision if features need to be cut from the plan to meet the release.

1.3.2.1 Release

Release for a particular product is normally scheduled for the end of a phase. However, circumstances may force a release out of cycle. When management agrees upon release of a product, the end of the third sprint in a phase is reserved for feature cutoff. No CRs or PBIs may be introduced to the product during the last sprint of a phase—the third sprint is reserved for bug fixes and integration testing.

Once all development is accomplished, a CCB meets and reviews all aspects of the product design, implementation, and vulnerabilities. After voting, if the CCB approves the product for release, the Director of Application Engineering posts the appropriate documentation and installers on the R:

(Release) drive. Every current product is located on the R: drive, and nothing can be delivered to a customer unless it comes from the R: drive.

1.3.2.2 Status

To ensure that the phase remains on track, metrics and reports are made available to management during the weekly directors meeting. The Operations Director presents the status of the phase plan from a CR perspective. The Director of Application Engineering presents the status of the current sprint from a PBI perspective, reports on trend lines, discusses other development issues, and presents outstanding bugs.

1.3.3 Review

Success of any process revolves around the ability to review what happened and how the team can should improve.

1.3.3.1 Phase Retrospective Meeting

On Day 1 (Wednesday) of the phase planning week, the Director of Application Engineering conducts a retrospective of the entire phase. During the retrospective, the team may opt to modify parts of the process that could be improved upon.

For more about this meeting, see Section A.1.9.

1.4 Sprints

While development revolves around the PBI, the overall WEdge process revolves around a sprint. Each sprint is three weeks long, starting on a Wednesday and ending on a Tuesday.

SPRINT WEEK	MONDAY	TUESDAY	WEDNESDAY	THURSDAY	FRIDAY
Week 0		(Previous Sprint Ends)	Day 1 • 0900 - Sprint Planning • 1300 - Sprint Planning Recap	Day 2 0900 - Directors Meeting, phase status, previous sprint report	Day 3 • 1300 - Customer Demo (from previous sprint)
Week 1	Day 4	Day 5	Day 6	Day 7 • 0900 - Directors Meeting, phase and sprint status	Day 8
Week 2	Day 9	Day 10	Day 11	Day 12 • 0900 - Directors Meeting, next sprint plan, phase and sprint status	Day 13
Week 3	Day 14 • 1700 - Code Cutoff	Day 15 • 1000 - Sprint Review • 1300 - Sprint Retrospective	(Next Sprint Begins)		

Figure 1–4 Sprint Calendar

1.4.1 Planning

The CR/PBI prioritization list presented at the last directors meeting is used for planning a sprint. This is the last opportunity to make changes to the list prior to sprint planning.

1.4.1.1 Sprint Planning Meeting

The first meeting of a sprint is the sprint planning meeting. This meeting determines which PBIs are going to be accomplished in the sprint and by which sprint team. During this meeting, the developers begin breaking down PBIs into initial SBTs. The PEs take ownership of this meeting.

For more about this meeting, see Section A.1.3.

1.4.1.2 Sprint Planning Recap Meeting

After the sprint planning meeting, the recap meeting is held as a review for the entire team. The overall plan for the next sprint is reviewed by each team PE.

For more about this meeting, see Section A.1.4.

1.4.2 Implementation

The following events occur while the sprint is underway.

1.4.2.1 Daily Scrum Meeting

The daily Scrum is held each work day during the sprint except on first and last day of the sprint. This meeting is a very quick commitment to team members about what they did yesterday and are going to do today, as well as a forum in which to raise anything in their way (impediments).

Any team member is welcome at the daily meeting, but only sprint team members speak. The PE allows 2 to 3 minutes to speak per sprint team member, to ensure a fast and effective meeting. During the daily meeting, no side conversations are to take place. Other meetings may result from this meeting, and the PE typically ensures that these follow-up meetings take place.

For more about this meeting, see Section A.1.1.

1.4.2.2 Development Methodology

WEdge develops against the user environment of Solutions Development Center (SDC) / Federal Desktop Core Configuration (FDCC), and unit tests are written for all code to the maximum extent possible. Additionally, code must be written to comply with coding standards and DoD guidance such as STIGs. Code is scanned using vulnerability scanning tools, and defects found in the scan are fixed. Section 3 presents further details about the development methodology.

The team has product development leads who own a specific function within WEdge. The product development lead must be aware of all code reviews and should attend whenever possible. Architecture of a new PBI is reviewed by the architect before beginning the code for the PBI.

1.4.2.3 Metrics

Every sprint must track metrics that are for management purposes. Sprint burndown charts are maintained during each sprint and are posted on the SharePoint portal. Other metrics such as trend lines and bug convergence predictions are presented to management during the directors meeting.

1.4.3 Review

Each sprint gets its own review to recap what happened and to potentially improve the process.

1.4.3.1 Sprint Review

On the last day of the sprint, the entire team assembles for a sprint review. In this meeting, the PE demonstrates each PBI that was accomplished during the last sprint. Feedback on features is encouraged, and this feedback is captured as CRs for the product.

For more about this meeting, see Section A.1.5.

1.4.3.2 Sprint Retrospective

The final meeting of a sprint is the sprint retrospective. Scrum teams meet with the Director of Application Engineering, who recaps the sprint and asks for any changes needed in the process.

For more about this meeting, see Section A.1.6.

1.4.3.3 Customer Demo

WEdge is very customer focused. Using Defense Connect Online (DCO), the operations team prepares a demonstration of the last sprint for customer feedback. This is the opportunity for customers to provide insight into their acceptance of the product before it is released.

For more about this meeting, see Section A.1.7.

2 WEdge Development Environment

The environment WEdge uses is key to keeping the process efficient. There are two elements to the environment: software and hardware

2.1 Software

WEdge currently uses assorted software applications to plan, develop, and provide metrics for the entire process.

2.1.1 Planning

The following software applications support planning.

2.1.1.1 Project

Project links directly with TFS 2010 for detailed CR and PBI planning and manipulation.

2.1.1.2 SharePoint

SharePoint provides realtime dashboards, drawn from TFS, for the management and planning of the development effort.

2.1.2 Development

The following software supports WEdge development.

2.1.2.1 Team Foundation Server

WEdge uses TFS as the cornerstone for development efforts. TFS stores, tracks, reports, and collates all development effort work items. This system also connects the Visual Studio development environment, to test suites, and allows non-developers access to all non-source code content, such as CRs, PBIs, and bugs.

WEdge uses two versions of TFS:

TFS 2008—used for legacy WEdge (1.0) and the Shuttle and Viewer releases under the Portable Flight Planning System (PFPS) Army umbrella, and cadet projects

TFS 2010—used for current WEdge (2.x) products

2.1.2.2 Visual Studio 2010 Ultimate

Visual Studio 2010 Ultimate is the preferred development platform for WEdge and is backward compatible to TFS 2008.

2.1.2.3 Visual Studio Team Edition 2008

Visual Studio Team Edition 2008 is an acceptable development platform for WEdge.

2.1.2.4 SharePoint

SharePoint allows for versioned document sharing and collaboration. With the incorporation of TFS 2010, all projects are automatically provisioned with their own website on the WEdge portal. These sites allow all interactivity with TFS. SharePoint also delivers TFS analytical processing metrics in charts. All CRs, PBIs, and bugs can be created, retrieved, updated, and deleted from this interface.

2.1.3 Metrics

In addition to the standard graphs and charts available through TFS, SharePoint, and Excel, WEdge uses other software packages for providing metrics, as follows.

2.1.3.1 Code Analysis

Code Analysis is a free, static code analysis tool from Microsoft that checks .NET managed code assemblies for conformance to the Microsoft .NET Framework design guidelines. Code Analysis analyzes the compiled object code, not the original source code. It uses Common Intermediate Language (CIL) parsing, and call graph analysis to inspect assemblies for more than 200 different possible coding standards violations.

WEdge uses Code Analysis design standards within the team code standards. When run, Code Analysis produces metrics on design violations that can be used to refactor code for better maintenance and reliability. Builds are rejected if a new (non-mitigated) warning occurs. In addition, cautions listed by builds must be identified and mitigated specifically as a Configuration Management (CM) responsibility for enforcement.

2.1.3.2 Fortify 360 and AppScan

ASACoE provides Fortify 360 for WEdge to generate security vulnerability reports on all products. WEdge also uses AppScan on databases to probe and report on vulnerabilities found. WEdge runs and reviews these reports every first directors meeting of the month and also with each software release, including service packs and hot fixes. Scans are also delivered to ASACoE for review every 90 days.

2.1.3.3 Performance Profiling

The Visual Studio premium profiling tools let developers measure, evaluate, and target performance-related issues in code. These tools are fully integrated into the Integrated Development Environment (IDE) to provide a seamless and approachable user experience.

Profiling an application is straightforward. The developer begins by creating a new performance session. In Visual Studio Team Edition for Developers, the developer can use the performance session wizard to create a new performance session. After a performance session ends, data gathered during profiling is saved in a .vsp file. The developer can view the .vsp file inside the IDE. There are several report views available to help visualize and detect performance issues from the data gathered.

2.1.3.4 Test Impact Analysis

By using Test Impact Analysis (available in Visual Studio 2010 Ultimate) during code development, the developer can identify the methods in a test project that have been affected by code changes in the managed code solution. At each build of the solution on the local computer, Test Impact Analysis identifies the methods in the code project that have changed and lists the test methods that directly call those methods. The developer can then run the tests from the Test Impact view window. The developer can also use the Test Impact View window to identify and run any test method that affects a particular code method.

Test Impact Analysis can also be used in Microsoft Test and Lab Manager, as part of the check-in policies for team projects in Team System, and in build definitions for Team Foundation Build.

2.1.3.5 Measuring Complexity and Maintainability of Managed Code

The increased complexity of modern software applications also increases the difficulty of making the code reliable and maintainable. In recent years, many software measures, known as code metrics, have been developed that can help developers understand where their code needs rework or increased testing.

Developers can use Visual Studio Team System to generate code metrics data that measure the complexity and maintainability of their managed code. Code metrics data can be generated for an entire solution or a single project.

2.2 Physical Hardware

The WEdge Scrum process employs the following physical devices and systems.

2.2.1 Scrum Board

The Scrum board is a visual meeting place where the team can facilitate relevant discussions and seek alignment about daily commitments. All PBIs, SBTs, and bugs are printed and applied to a magnetic white board to make up the working Scrum board. The board represents the current development effort in any given sprint.

> **Note**
> Even though the team uses a physical white board for most discussions, the TFS system should be regarded as the Gold copy for work item information.

PBIs remain at the top of the board, and SBTs move through the various states of completeness (see Figure 2–1). Bugs move from top to bottom through the same progression as SBTs. The board is divided into the following sections:

- Not Done
- In Progress
- Ready for Review
- Ready for Test
- Done

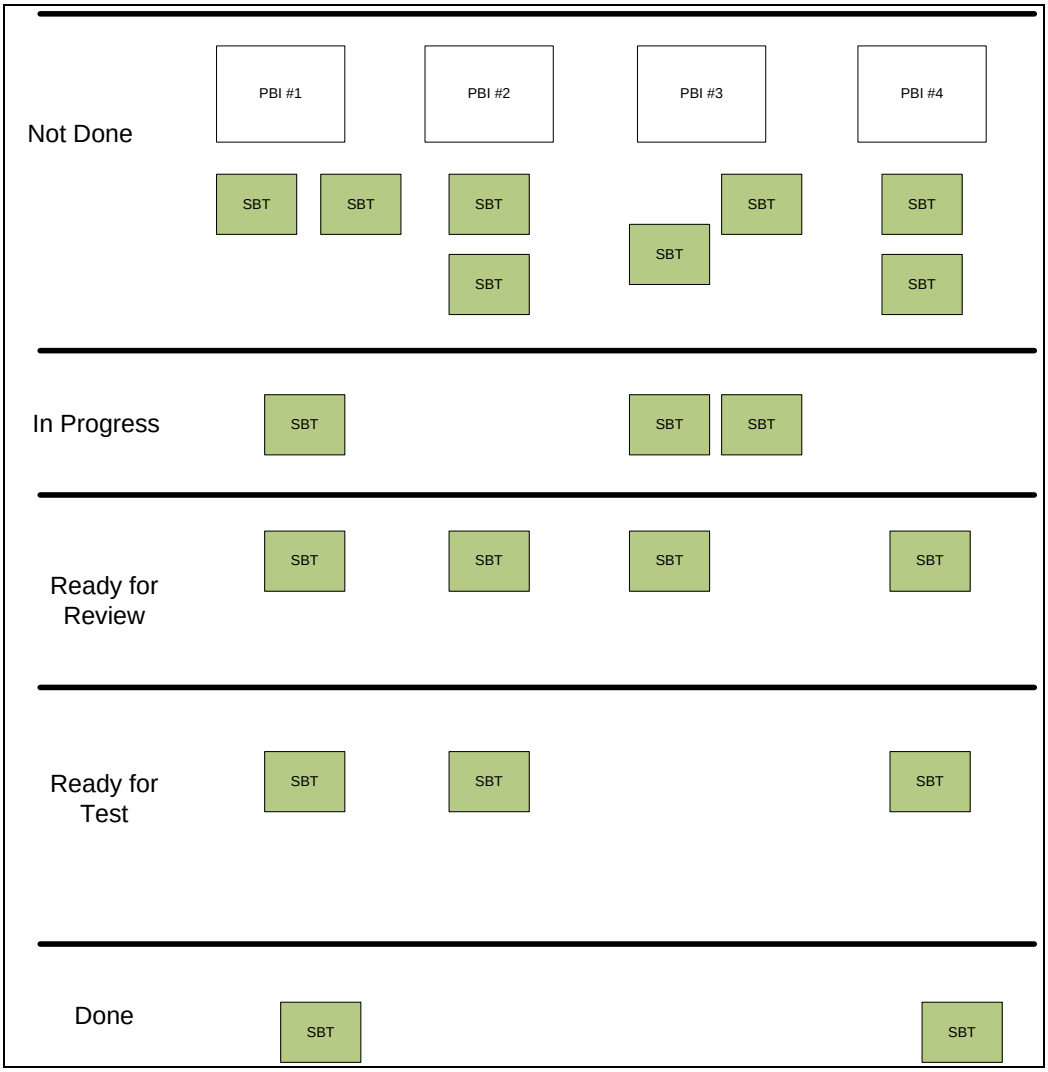

Figure 2–1 Scrum Board Example

2.2.2 Servers

WEdge uses powerful servers running Windows Server 2008 with Hyper-V virtualization for development and test environments, as well as for running all administrative support structure (e.g., Exchange server, domain controller). Also, the operations directorate has virtual environments in order to provide training and support separate from any development or testing.

The development virtual environments and test virtual environments are run separately but may "spin up" common virtual machine environments so that bugs found in test may be debugged in the exact same environment in which they were found.

Visual Studio Lab Management creates and manages virtual environments on a pool of Hyper-V hosts and System Center Virtual Machine Manager library servers. Using these environments, Lab

Management, TFS, Team Test Load Agent, and Team Test Load Controller provide an integrated software testing experience that lets testers attach comprehensive snapshots of the environment to the associated bugs. Lab Management also lets the team automate the authoring and running of end-to-end application lifecycle management workflows.

2.2.3 Magnetic White Boards

WEdge uses numerous white boards for design sessions. Some of the boards can scan their surface so that notes may be captured as written and then be attached to CRs, PBIs, or SBTs. All white boards support magnets so that physical cards may act as an input source.

3 WEdge Development Lifecycle

WEdge uses the following items to track and report development efforts:

- **Change Request** (CR)—A product idea that spans one or more sprints
- **Product Backlog Item** (PBI)—A product requirement, a piece of a CR that can be completed within one sprint
- **Sprint Backlog Task** (SBT)—A small task that must be accomplished in order to complete a PBI
- **Impediments**—Anything that prevents a PBI from being worked
- **Acceptance Test** (AT)—An SQA test that maps directly to a PBI CoA

Figure 3–1 and Figure 3–2 show the overall flow of development, from idea through demonstration and release.

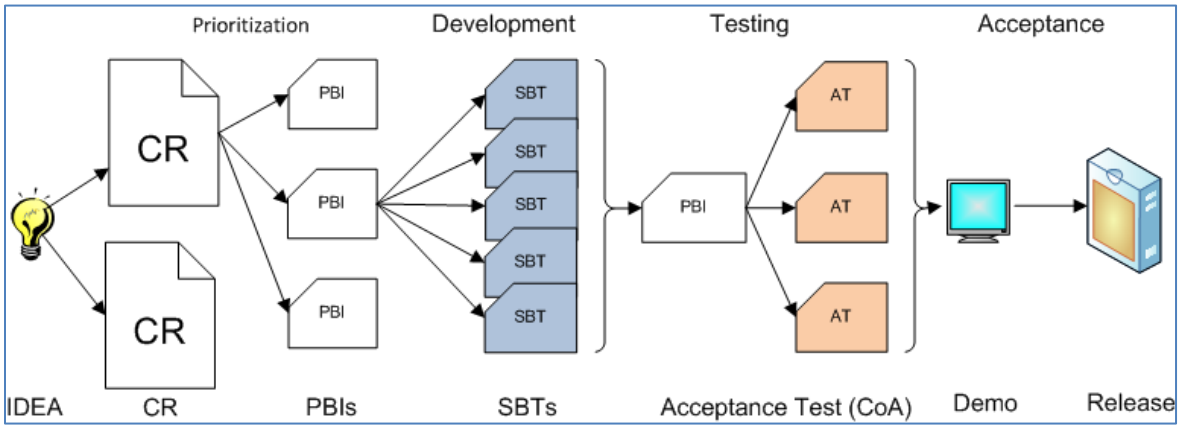

Figure 3–1 Idea-to-Release Process

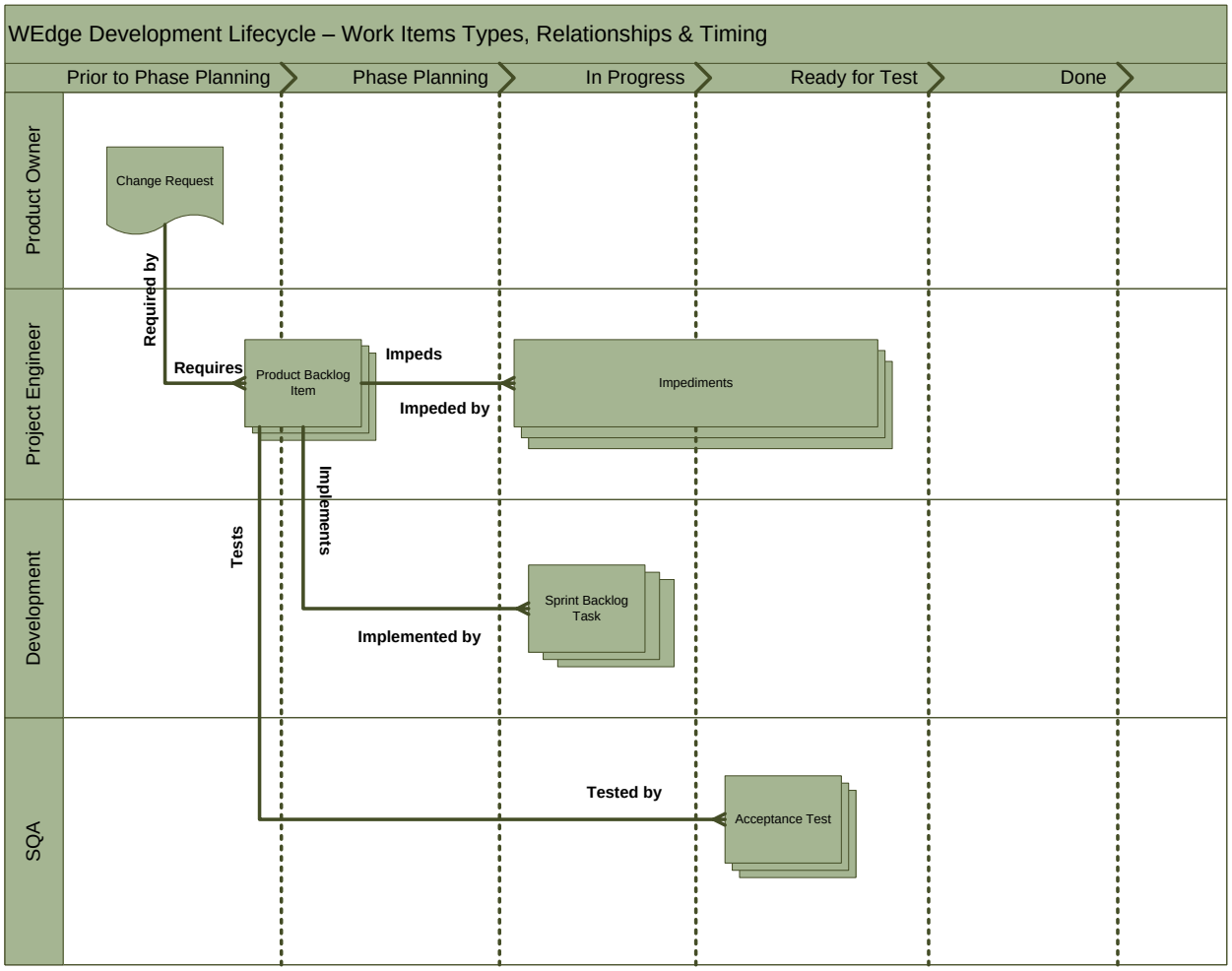

Figure 3–2 Lifecycle Swim Lane Diagram

3.1 Change Requests

Development begins with ideas, and an idea may be captured and submitted by anyone as a new CR. CRs include a title and description, plus a section to capture the overall goals of the idea. When the goals are met, the idea is considered fully realized, and the CR is Done. CRs are written as work items in TFS. Metadata describing the CR includes:

- **Product Owner** (a member of the operations team)
- **Author** (who originated the idea—if outside the WEdge team, this is the PO as proxy)
- **Product** (e.g., WEdge Briefing Client, Shuttle, Viewer, WEdge Master, Plug-in)
- **Product Area** (e.g., Sync, My Briefs, Shuttle Buckets)
- **Priority** (where this CR ranks among all other CRs)

- **Current Status** (e.g., Not done, In Progress, Done)
- **Deployed with Release** (the release containing the final CR PBIs)

CRs follow this progression:

Not Done → In Progress → Done

Newly created CRs begin in the Not Done state. Once PBIs are linked to the CR, it is moved to In Progress. The PO of a CR has complete responsibility for that CR, and only the PO may move a CR to Done status.

The POs of each product are mindful of the overall WEdge vision when combining CRs from all products into one prioritized list. This list can be viewed at any time and by anyone in order to see the overall direction of WEdge. The CR list is also be reviewed by the WEdge Director team each week. The WEdge Director has the final authority over the list.

A CR is considered Done when:

- All of its PBIs are in Done status
- All its goals are met
- It is signed off by its PO

3.2 Product Backlog Items

PBIs describe the full implementation of, and are the business requirements in, a CR. They are individual features that can be coded, tested, and demonstrated within one sprint. PBIs include a title, and a description that captures the requirement.

All PBI creation involves the architect and the product development lead to ensure requirements are feasible with current WEdge design. As such, the architect may add implementation thoughts to the PBI. All documents associated with the design of a PBI are attached to the PBI item in TFS, including any hardcopy notes (which must be scanned in). Metadata describing the PBI includes the following required fields:

- **Projected Sprint** (for planning purposes)
- **Completed Sprint** (the sprint in which the PBI was marked Done)
- **Estimated Effort** (for planning purposes and is only the relative "bigness" of a PBI)
- **Scrum Team**
- **Priorit**y (where this PBI ranks among all other PBIs)
- **Assigned To**
- **Work Remaining** (total of all associated SBT hours)
- **Current Status** (Not Done, In Progress, etc.)
- **Product** (e.g., WEdge Briefing Client, Shuttle, Viewer, WEdge Master, Plug-in)

- **Product Area** (e.g., Sync, My Briefs, Shuttle Buckets)
- **Project Engineer** (the PE responsible for requirements and status of this PBI)

PBIs follow this progression:

Not Done →In Progress →Ready for Review →Ready for Test →Operations Review →Done

Newly created PBIs default to Not Done. When a developer begins work on the PBI for a sprint, the developer moves it to In Progress and self-assigns it. After all tasks to complete the PBI are done, the developer moves the PBI to Ready for Review, and a code review is conducted IAW WEdge code standards. Once the code review is complete, the developer who ran the review moves the PBI to Ready for Test and assigns it to the SQA lead.

SQA tests PBIs against the CoA first. Any bugs created when ATs fail must be fixed before the PBIs is Done. If SQA finds bugs against the CoA, the tester moves the PBI back to In Progress and assigns it back to the original developer. Once there are no open bugs against the CoA, the tester moves the PBI to Operations Review status and assigns it to the Operations Director.

PEs are responsible for the content of PBIs and the prioritization of the PBI list. The PBI list priority must support the priority order of CRs.

Any time an event or issue impedes work on a PBI an Impediment is created by the PE responsible for the PBI.

A PBI is considered Done when:

- All of its SBTs are Done
- All of its Impediments are Done
- All of its ATs are Passed

3.3 Sprint Backlog Tasks

SBTs are any tasks (coding or otherwise) that must be done in order to meet a PBI CoA. SBTs include a title, and a description that captures the details of the task being completed. SBTs should be small enough to be completed in one work day (no more than 5 hours). Metadata describing the SBT includes the following required fields:

- **Completed Sprint** (this is the sprint in which the SBT was completed)
- **Remaining hours** (these hours roll up into the overall burndown charts)
- **Current Status**

SBTs follow this progression:

Not Done →In Progress →Done

3.4 Impediments

Impediments are ***any*** issues that get in the way of completing PBIs. Examples of impediments include network outages, software application problems, unforeseen personnel issues, or even the lack of experience and knowledge of the techniques necessary to implement a PBI. Impediments include a title, and a description that captures the details of the impediment and possible mitigations and solutions. Metadata describing the Impediment includes the following required fields:

- Current Sprint (the sprint where the impediment was identified)
- Project Engineer (the PE who is responsible for removing the impediment)

Impediments follow this progression:

Not Done →In Progress →Done

3.5 Acceptance Tests

ATs are test scenarios specifically written to test PBI CoA, which are listed in TFS and linked directly to PBIs. PBIs cannot be closed until all ATs are Passed. ATs include a title, and a description that captures the test steps and pass/fail criteria. Metadata describing an AT includes:

- Tester (the SQA personnel responsible for the test)
- Current Sprint

ATs follow this progression:

Not Done →In Progress →Passed/Failed

If an AT fails, details of the failure are captured in the AT and immediately addressed by the tester and the developer of the associated PBI. They resolve the failure, and rerun the test, repeating as necessary until the test passes.

3.6 Software Bugs

WEdge software bugs are addressed through the following processes.

3.6.1 Bug Discovery

Bug discovery is handled in one of two ways:

Bugs found during development in relation to an open PBI

Bugs discovered by SQA (or anyone) during sprint testing of a PBI that has not been marked closed are set back to In Progress and returned to the developer to fix the bug. Developers or testers do not need to “write up” a formal bug for these found issues, unless necessary to properly describe the problem to development.

Bugs unrelated to currently open PBIs

Bugs that are discovered at any time outside of a sprint, or during a sprint when the PBI has been closed, must follow the bug process described in the next section.

3.6.2 Bug Process

Bugs are entered in TFS and can be submitted by anyone. Submitted bugs must include enough information so that SQA can follow the problem. SQA reviews all new bugs and ensures that they are not duplicates, and that they can be reproduced or verified.

POs prioritize verified bugs into one of the following categories:

Critical—The bug causes a system shutdown or blue screen, or it halts testing. This status also applies when a customer finds an issue in the field and it prevents them from using the product.

High—The bug causes a serious, non-critical behavior that is unacceptable for release.

Medium—The bug causes an undesirable behavior but is acceptable for release.

Low—The bug is a minor annoyance but is acceptable for release.

At any time, the PO can reprioritize bugs as necessary. Any bugs moved into High or Critical must be discussed with the Director of Application Engineering, and presented at the next directors meeting.

3.6.3 Implementation of Bug Fixing

Critical bugs are assigned to two developers, who stop all work on the current sprint and fix the critical bug. These developers are assigned by the development manager.

High bugs are assigned to sprints as bug items and treated like a PBI. They should be estimated like all PBIs.

Medium and Low bugs are estimated but not treated like PBIs. They are assigned during the fourth (last) sprint of a phase as appropriate.

Product Releases: All open bugs in previous release notes are AUTOMATICALLY moved to High for the next phase. During the last sprint of a phase before a product release, bugs are worked in order of severity—all Critical and High, then Medium, and finally Low.

Bugs follow this progression:

Not Done →SQA Confirmed → PO Prioritization → In Progress →Ready for Review → Ready for Test →Ops Review → Done

A Bug being worked during a Sprint that is unrelated to a current PBI should be compared to the original requirement to ensure proper operation and meet any listed conditions of acceptance. If this Bug fails testing to the COA criteria, it should be moved by SQA back into Not Done status and unassigned from any developer. This allows visibility into currently open bugs and allows any developer on the team to identify and address that bug. Any failed testing notes should be included in the History section of that Bug work item in TFS. Also, any changesets checked in to

address this bug should be linked to the Bug work item in TFS and sufficient check-in notes provided to allow the progression of work to remediate the bug to be easily understood.

3.7 Software Vulnerability Testing

WEdge employs software engineers who are CEHs of systems. Once certified, these developers become part of the WEdge Red Team while still being part of normal Scrum teams. Their Red Team purpose is to identify and exploit security vulnerabilities in WEdge.

When vulnerabilities are found, red PBIs may be created by the Red Team and then implemented through the normal process. A red PBI must be worked at least once every 90 days, per product. Also, unplanned red PBIs may be injected at the sprint planning meeting by the WEdge Director.

3.8 Emergency Process Modifications

There are always emergencies. With that in mind, minor immediate modifications to the processes described in this document are acceptable. However, they must be cleared through the Director of Application Engineering or WEdge Director. Major modifications or deviations from this process must be brought to the attention of the WEdge Director and presented for clarification and decision.

3.9 Hot Fixes

Hot fixes are worked in their own branch. After passing CCB, hot fix modifications are added to the development branch source code for inclusion in the next release.

Appendix A ▪ Detailed Meeting Plans

This appendix lists the regular and unscheduled meetings that support the WEdge Scrum process.

> **Note**
> All WEdge meetings begin on time, with time pieces synchronized to the Naval Observatory Master Clock at DSN (94–) 762–1401.
> Required attendees who are late to any scheduled meeting are assessed a $1.00 foul, collected by the meeting owner.

A.1 Regularly Scheduled Meetings

Regularly scheduled meetings are required during various parts of the sprint process, as follows. They are held at a predictable, recurring day and time.

A.1.1 Daily Scrum

PURPOSE:	To commit ownership of tasks to the Scrum team and identify impediments
MEETING TIME:	Approximately 8:30 AM (0830) daily, excluding the first and last days of a sprint. Actual time is at the discretion of the PE.
OWNER:	Scrum Team Project Engineer
REQUIRED ATTENDEES:	Scrum Team
OPTIONAL ATTENDEES:	Anyone

The Scrum team meets almost every morning. Only Scrum team members can actively talk during this meeting, until the very end. The PE keeps the meeting on track and asks for comments from non-members at the end. This is a very structured meeting so that it can be kept short.

Three questions are answered by each speaker:

1) What did you do yesterday?

2) What are you going to do today?

3) What is keeping you from accomplishing tasks?

The meeting is a commitment to the team but often sparks discussion. While these discussions are vital to continued development, they are best postponed until a follow-up meeting. The PE ensures that all discussions revolve around the three questions, and that further discussion is tabled to follow-up meetings. Developers physically move SBT cards on the Scrum board from one state to another. Every developer should move at least one card, or the SBTs are tasked too broadly.

Any observers (non Scrum team members) should be addressed at the end of the meeting for their own follow-up meeting requests.

All impediments identified from this meeting are documented and handled by the PEs, and elevated to management if necessary.

Late Scrum team members are assessed a $1.00 foul, while unexcused absences are a $2.00 foul. Traveling Scrum team members should call in to the daily meeting if possible. The PE ensures that a conference phone is available from 2 minutes prior to start, to one minute after. If no one calls in within that time, the line is hung up.

The current call-in number is: (719) 333–0140

A.1.2 Weekly Directors Meeting

PURPOSE:	To provide a common communication foundation for the team, inform the WEdge Director of events and problems, solve issues, and assign tasks
MEETING TIME:	9:00 AM (0900) on every Thursday or as scheduled
OWNER:	WEdge Director
REQUIRED ATTENDEES:	WEdge Director and department directors
OPTIONAL ATTENDEES:	The meeting is usually closed; however, others may be invited to provide special information to complete the decision making process.

This meeting is owned by the WEdge Director, and items for inclusion on the agenda are due by COB Wednesday.

This is a closed meeting where team direction and decisions are made. Each director provides slides for this meeting. Meeting minutes are recorded. A meeting recap is posted on the SharePoint site within 24 hours. Each director should have an informational sub meeting with their team as appropriate.

The prioritized CR and PBI lists are presented at this meeting for approval and adjustment, and are then passed down through appropriate directors. Status of the phase and current sprint are also highlighted.

A.1.3 Sprint Planning Meeting

PURPOSE:	For sprint teams to select PBIs that they commit to completing in the sprint, and to refine initial SBTs for work during the sprint
MEETING TIME:	9:00 AM (0900) on the first day of the sprint (Wednesday of Week 0).
OWNER:	Project Engineer
REQUIRED ATTENDEES:	Scrum teams, Director of Application Engineering, Testers, POs, Deployment Developer, Architect, Technical Writer
OPTIONAL ATTENDEES:	Anyone
INPUTS:	The prioritized PBI list. Complete list of team velocity compiled by the PE for their team. PEs are responsible for adequate supplies

	(index cards and markers)
OUTCOME:	SBTs along with the estimated number of hours required to complete the PBIs that the team has committed to accomplish. List of risks and mitigations.

A PE provides focus for the development teams at the beginning of this meeting. All PBIs that were not accomplished in the previous sprint are automatically moved to the top of the prioritized PBI list unless overridden by the Director of Application Engineering. Following pushed PBIs, Scrum teams select items from the prioritized PBI list on the planning board, starting with business priority #1 and concluding when no more velocity is available for another PBI from the list, OR if the team identifies there are no more hours available for work.

Scrum teams may, by exception and only when cleared with the Director of Application Engineering, select PBIs out of order, but only if they are related to or dependent on the PBI selected.

As PBIs are selected, developers read the full PBI, including initial design elements and CoA. Figure A–1 shows a sample PBI card.

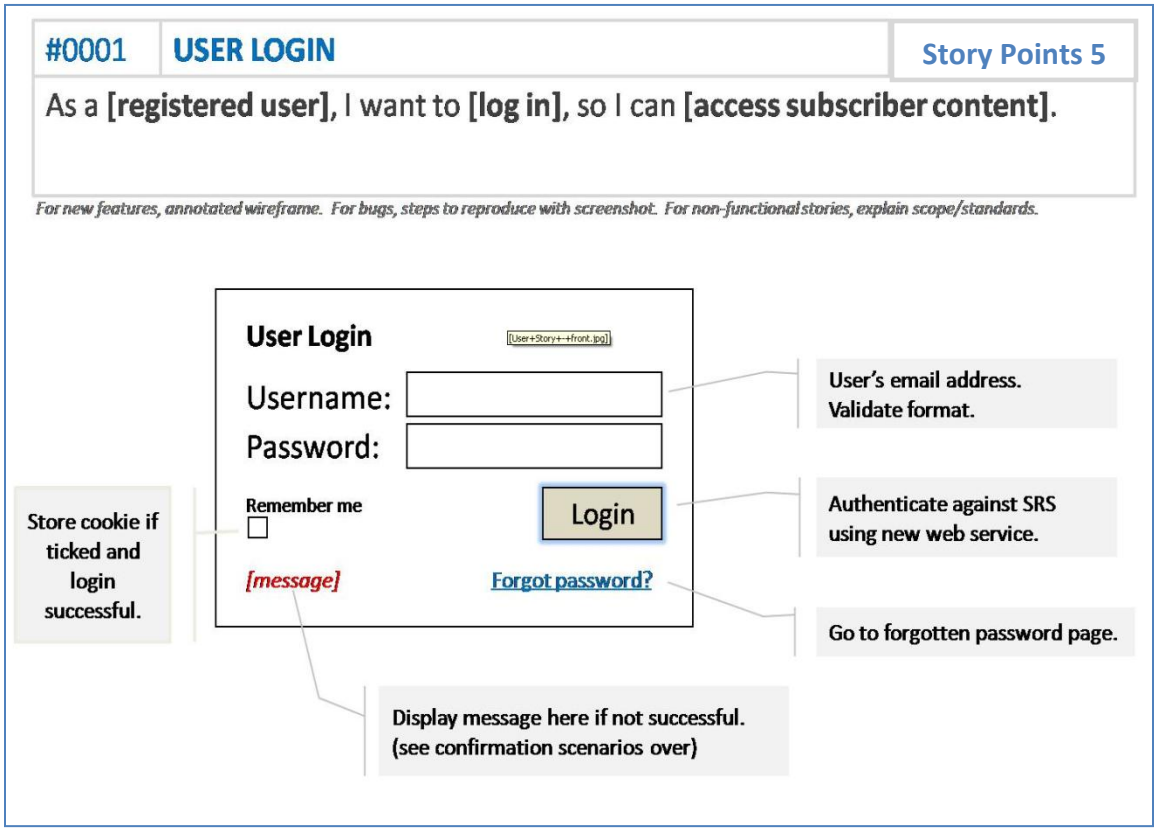

Figure A–1 Sample PBI Card

PBIs are broken into full design and workable tasks (less than 5 hours) called SBTs. SBTs are created for each PBI until all tasks are identified. Developers are responsible for full design implementation

of the PBI, code investigation prior to development, and unit testing. To fully understand the PBI, concept discussions with the PO, code discussions with the product development lead and architect, CoA questions, and identification of needed research tasks are considered.

SBTs are marked with the PBI number in the upper right and the number of hours to accomplish the task in the lower left. See Figure A–2.

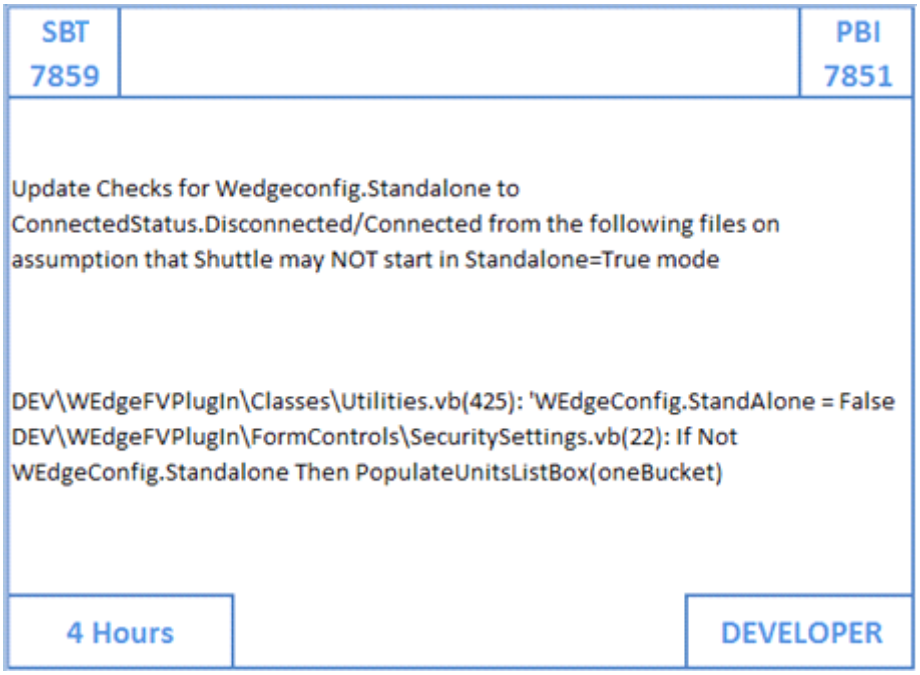

Figure A–2 Sample SBT Card

The PE is responsible for keeping track of SBT time and the remaining hours of availability for their team. Figure A–3 shows an example Excel spreadsheet for this task, but PEs have the flexibility to use their own method of tracking.

The team may decide to plan tasks for additional PBIs that may be added to the task board later in the sprint, if they have completed all committed work and can commit to more.

Team Availability Hours for Upcoming Sprint

Phase 10.1 **Sprint 2**

Team: **Atlantis**

		Team Member											
		Developer 1		Developer 2		Developer 3		Developer 4		Tester		PE	
		Hours	Comment	Hours	Comment	Hours	Comment	Hours	Comment	Hours	Comment	Hours	Comment
Week 1	Day 1	0	PTO	5		5				2	Other	5	
	Day 2	5		5		5				2	Other	5	
	Day 3	5		5		5				2	Other	5	
	Day 4	5		5		5				2	Other	5	
	Day 5	5		5		5				2	Other	5	
Week 2	Day 6	5		5		3	PTO			5		5	
	Day 7	5		5		5				5		5	
	Day 8	5		5		5				5		5	
	Day 9	5		5		5				5		5	
	Day 10	5		5		5				5		5	
Week 3	Day 11	0	Trip	5		0	Trip			5		5	
	Day 12	0	Trip	5		0	Trip			5		5	
	Day 13	0	Trip	5		0	Trip			5		5	
	Day 14	5		5		5				5		5	
	Day 15	5		5		5				5		5	
Total Developer Available Hours		**55**		**75**		**58**		**0**		**60**		**75**	
Total Team Available Hours		**323**											

Figure A–3 Sample Hours-Available Sheet

Risk Identification—At the sprint planning meeting, the team identifies risks that might prevent completion of the committed work. A developer out of the office is not a risk, as that should be considered before pulling a PBI. Risks should be fairly rare, but if present, must be assessed for impact and probability. Mitigation plans are included for each risk identified. The mitigation plan will be inserted as an Impediment work item type and identified in the title as a Sprint Risk. Risks will be worked as impediments and closed at the end of the Sprint even if the risk never came to be.

A.1.4 Sprint Planning Recap Meeting

PURPOSE:	Summarize the commitments made during planning to the team. Commitments and risks can be made visible to management.
MEETING TIME:	1:00 PM (1300) on the first day of a sprint (Wednesday of Week 0)
OWNER:	Director of Application Engineering
REQUIRED ATTENDEES:	Scrum teams, Directors
OPTIONAL ATTENDEES:	Anyone
INPUTS:	Complete PBIs broken into SBTs per Scrum team
OUTCOME:	Team understanding of the upcoming sprint and risks associated

with it. SBTs and PBIs are updated in TFS by the PEs.

This recap meeting is held to summarize the commitments of each sprint team on the PBIs they have committed to accomplish during the sprint. Each PBI is reviewed and an overview of each SBT occurs. Any PBIs that the team has questions or comments about are brought up at the recap meeting. If a PBI is at risk for accomplishment, that should be highlighted to the team.

At the conclusion of the meeting, developers enter SBTs into TFS, but PEs are responsible for ensuring that everything was input accurately and completely.

A.1.5 Sprint Review Meeting

PURPOSE:	Provides a critical checkpoint for the development team to demonstrate the accomplishments of the sprint, and to ensure that all requirements and CoA were met and are acceptable to the PO
MEETING TIME:	10:00 AM (1000) on the last day of the sprint, Tuesday of Week 3
OWNER:	Project Engineers
REQUIRED ATTENDEES:	Entire WEdge team
OPTIONAL ATTENDEES:	Anyone
INPUTS:	Complete list of PBIs completed during the last sprint. Demonstrable conditions for each PBI practiced.
OUTCOME:	Team understanding via demonstration of the development accomplishments. CRs may be generated if CoA was met but POs or directors are not happy with the outcome. A list of customer demonstrable PBIs for the customer demo.

During the review, PEs demonstrate sprint accomplishments to the team, using a neutral demonstration box (i.e., not a development box). Normally, the entire team is available for a sprint review, and preparation is key for this event.

Any observations may be made, and comments are encouraged. Any work that needs to be accomplished that is outside the CoA becomes a CR for future sprints.

The meeting is dismissed with a reminder to accomplish any end-of-sprint checklists.

A.1.6 Sprint Retrospective Meeting

PURPOSE:	Evaluate the Scrum process and identify opportunities for improvement
MEETING TIME:	1:00 PM (1300) on the last day of the sprint, Tuesday of Week 3
OWNER:	Development Manager
REQUIRED ATTENDEES:	Scrum teams, Director of Application Engineering
OPTIONAL ATTENDEES:	Anyone
INPUTS:	Items and issues identified from the last sprint retrospective
OUTCOME:	Identify 1) What worked well in the sprint

2)What needs to be changed in the next sprint
3) What items need to be watched for further consideration

Any open issues identified from previous retrospectives that have not been resolved are highlighted and resolved prior to the end of the next sprint.

Each person in the meeting receives cards on which to write items that went well or that they would like to see changed. Each person posts their cards on a white board with magnets, either on the "plus" or "minus" side of the board. Post accordingly.

All items are reviewed by the development manager for discussion with the team. The Director of Application Engineering brings the latest copy of this document to all retrospective meetings to annotate any items that dictate a change in the process.

A.1.7 User Demo

PURPOSE:	To share with the user community what WEdge has been working on and obtain customer feedback
MEETING TIME:	1:00 PM (1300) on the Friday following the end of a sprint
OWNER:	Operations Director
REQUIRED ATTENDEES:	POs, Operations Director
OPTIONAL ATTENDEES:	Anyone
INPUTS:	Working demonstration, rehearsed. Active DCO connection
OUTCOME:	Feedback and acceptance on new features and functionality from customers

Customers are notified about upcoming demos approximately one week ahead of time, about the time of the demo and its potential content. Wedge team members do not demonstrate items from the sprint review that failed, unless fixed and fully able to be demonstrated. The primary demo focus is on customer needs and to ensure that the dialog is customer centered. The demos are a vital part of the WEdge process.

Connect to the WEdge Demonstration room in DCO at https://connect.dco.dod.mil/WEdge. Record the session, and store it for others to view, should they miss the meeting. Record all inputs from customers and create CRs as appropriate.

A.1.8 Phase Planning Meeting

PURPOSE:	To create a plan and desired outcome for the next phase
MEETING TIME:	1:00 PM (1300) on the first day of the phase week (Wednesday)
OWNER:	Director of Application Engineering
REQUIRED ATTENDEES:	WEdge Director, Director of Application Engineering, POs, PEs, Architect, Development Manager
OPTIONAL ATTENDEES:	Anyone
INPUTS:	Prioritized CR implementation, per product, during the next phase. For each CR, estimated PBIs are listed for desired implementation. Combined running average velocity for all Scrum teams per sprint.

OUTCOME:	Completed CR/PBI plan covering the next phase

POs and PEs collectively meet to discuss what CRs they would like to accomplish in the next phase. They present these CR lists, per product, to the directors during a weekly directors meeting, well before the phase planning meeting.

All CRs must have their estimated PBIs (requirements) before they can be considered for inclusion in the phase. It is the responsibility of the PE of the product to ensure that CRs are broken up into PBIs and that each PBI has been appropriately estimated.

At the beginning of the meeting, the Director of Application Engineering reviews the list of CRs that the POs want for the next phase. Each product has a priority list of CRs. If a CR is too big to fit into the phase, it is identified before discussion.

The velocity per sprint is written on a master planning document. Each CR, in order of importance, is placed on the planning document. PBIs required to complete the CR are placed and planned per sprint to ensure completion as necessary. This process continues until velocity limits are reached.

If a product is identified for release, no PBIs may be included in the fourth sprint of a phase.

> **Note**
> Phase planning does not preclude the ability to reprioritize before a sprint, nor does it carve things in stone. It is a plan only; it should be followed as closely as possible, but it is not locked down.

A.1.9 Phase Retrospective Meeting

PURPOSE:	Review phase overall implementation. Evaluate the Scrum process and identify opportunities for improvement.
MEETING TIME:	9:00 AM (0900) on the first day of the phase week (Wednesday)
OWNER:	Director of Application Engineering
REQUIRED ATTENDEES:	Directors, Development Manager, PEs
OPTIONAL ATTENDEES:	Anyone
INPUTS:	List of all retrospective items from all sprints. Any concerns to raise.
OUTCOME:	A full lessons-learned evaluation of the phase. A list of 1) items to improve upon, 2) do better and 3) keep doing.

The phase retrospective consists of three events.

Sprint Retrospective Review—The phase retrospective meeting owner brings a list of all issues identified from the phase sprint retrospectives. Each item is reviewed in terms of how it was handled and its capabilities.

Phase Process Review—Much like a sprint, each person in the meeting receives cards on which to write items that went well or that they would like to see changed. Each person posts their cards on a “plus” and “minus” white board with magnets. Post accordingly. All items are

reviewed with those present. Any items that dictate a change in process are elevated to the Director of Application Engineering and implemented appropriately.

General Open Forum—Any issues or concerns, about anything, may be raised and discussed.

A.1.10 Configuration Control Board Meeting

PURPOSE:	To approve product release and ensure DoD policy is followed
MEETING TIME:	10:00 AM (1000) on the last day of phase week (Tuesday) or as necessary
OWNER:	Network and Security Director
REQUIRED ATTENDEES:	CCB members
OPTIONAL ATTENDEES:	CCB is usually a closed meeting; however, others may be invited to provide special information to complete the CCB process.
INPUTS:	Source code, release notes, version description document, ASACoE scans, IAVA compliance
OUTCOME:	Approval or denial for release

The meeting owner compiles artifacts at least two days prior and emails them to CCB members. The Director of Application Engineering provides source code at the meeting. This is a formal process run IAW CCB guidance, ending with a formal approval and archiving of artifacts. The release manager (a specific duty assigned by the WEdge Director) moves all artifacts to the R: drive and archives them on the Z: drive appropriately.

A.2 Unscheduled Meetings

The following meetings are held as needed.

A.2.1 Planning Estimation Meeting

PURPOSE:	To give a gut feel estimation for each PBI not estimated
MEETING TIME:	As called
OWNER:	Development Manager
REQUIRED ATTENDEES:	Developers, PEs, POs
OPTIONAL ATTENDEES:	Anyone
INPUTS:	List of PBIs not yet estimated
OUTCOME:	Enough PBIs estimated to provide planning for the next sprint or phase

PEs are responsible for ensuring that enough PBIs are estimated for the next event. The PE pulls the first PBI that needs estimating and gives a verbal description of the PBI to the developers. The developers then estimate the PBI on the following scale from “smallest” to “biggest” as follows:

1, 2, 3, 5, 8, 13

The developers discuss differences in the numbers assigned to each PBI, and then vote again. As many rounds of estimation as necessary are run until the development team comes to a consensus

on the scale number of each PBI. Any PBI that the development team thinks is beyond 13 goes back to the PE to be broken into more PBIs, with suggestions from development. Developers are free to clarify any information in PBIs to help in their estimation.

A.2.2 CR/PBI Prioritization

PURPOSE:	To discuss prioritization of items for the next phase or sprint
MEETING TIME:	As required
OWNER:	Product Owner
REQUIRED ATTENDEES:	Product Owner, Project Engineers
OPTIONAL ATTENDEES:	Anyone
INPUTS:	List of desired PBIs and CRs for the next sprint or phase
OUTCOME:	A prioritized list of PBIs and CRs for the next sprint or phase

The prioritization of product features is a continually updated list. At each directors meeting, a myriad of metrics are discussed. One of them is the current CR priority list and current PBI priority list. The POs and PEs must get together and agree on the lists per product.

These lists are different for the PBIs and CRs implemented in a sprint only by the fact that the sprint list is a combined PBI list for all products that must be combined before the sprint planning meeting.

Appendix B ▪ WEdge Products

This appendix lists the WEdge products that are managed through the Scrum process described in this document.

- WEdge Repository Server (Tier 1)
- WEdge Master Server (Tier 2)
- WEdge Briefing Client
  - Software Development Kit (SDK)
  - Patriot Excalibur (PEX) Plug In
  - Notices to Airmen (NOTAMS) Plug In
- WEdge Shuttle
- WEdge Viewer
- Web Site
- Range Application
- WEdge Team Utilities
- WEdge Common (any DLL or EXE files used by more than one other product)

Appendix C ▪ Terminology

The following terms appear in this document or are otherwise associated with the WEdge effort.

Table C–1 Terminology

Term	Definition
ACC	Air Combat Command
ANG	Air National Guard
ASACoE	Application Security Assurance Center of Excellence
AT	Acceptance Test
C&A	Certification and Accreditation
CCB	Configuration Control Board
CEH	Certified Ethical Hacker
CIL	Common Intermediate Language
CM	Configuration Management
CoA	Conditions of Acceptance
COB	close of business
CR	Change Request
DCO	Defense Connect Online
DLL	dynamic link library
DoD	Department of Defense
DSN	Defense Switched Network
ECM	Enterprise Content Management
EXE	executable
FDCC	Federal Desktop Core Configuration
IAM	Information Assurance Manager
IAO	Information Assurance Officer
IAVA	Information Assurance Vulnerability Alert
IAW	in accordance with
IDE	Integrated Development Environment
NOTAMS	Notices to Airmen
PBI	Product Backlog Item
PE	Project Engineer
PEX	Patriot Excalibur
PFPS	Portable Flight Planning System
PMR	Project Management Review
PO	Product Owner
SBT	Sprint Backlog Task
SDC	Solutions Development Center
SDK	Software Development Kit
SQA	Software Quality Assurance
STIG	Security Technical Implementation Guide
TFS	Team Foundation Server
WEdge	Warfighter's Edge

This page intentionally left blank